Procesamiento del Lenguaje Natural, Revista nº 61, septiembre de 2018, pp. 197-200 recibido 05-04-2018 revisado 26-04-2018 aceptado 04-05-2018

QuarryMeaning: Una aplicación para el modelado de tópicos enfocado a documentos en español

QuarryMeaning: A Topic Model Application focused on Spanish Documents

Olga Acosta, César Aguilar y Fabiola Araya
Facultad de Letras de la Pontificia Universidad Católica de Chile
Campus San Joaquín, Santiago de Chile
{oacostal;caguilara;fbaraya}@uc.cl

Resumen: Esta demostración presenta una aplicación *standalone* que permite entrenar y probar un modelo de tópicos. Tal aplicación considera filtros para reducir *ruido* en los resultados. Así, por una parte, se incluye una lista de palabras base no relevantes que se puede complementar con otros vocabularios, ya sean propuestos por el usuario, o bien obtenidos mediante un enfoque comparativo usando un corpus de referencia. Por otro lado, es posible considerar únicamente las palabras que tienen un valor semántico alto usando etiquetas de partes de la oración. Además, se incluye un despliegue visual de nubes de palabras que muestra los primeros 10 tópicos derivados del entrenamiento, con el objetivo de explorar visualmente los resultados. Finalmente, se realizó la evaluación de la herramienta considerando una tarea de clasificación de documentos. El modelo logró niveles de precisión superiores al 95% en el conjunto de prueba.
Palabras clave: Procesamiento de lenguaje natural, minería de textos, modelación de temas, enfoque contrastivo, clasificación de textos

Abstract: This demo shows a standalone application that allows to easily train and test a topic model. The application includes filters for reducing noise in the results. On the one hand, a base stop-list is included, but it can be complemented with a non-relevant word list proposed by user, or obtained it by means of a contrastive approach using a reference corpus. On the other hand, words having a high semantic value can be considered using POS tags. We also include a visualization in word-clouds way, where ten topics can be shown, in order to analyze in detail the results. Finally, evaluation was carried out focusing topic model for classifying documents. Our model achieved levels of precision above 95% in the test set.
Keywords: Natural language processing, text mining, topic modeling, contrastive approach, text classification

1 Introducción

En el análisis de datos, tradicionalmente, la reducción de dimensionalidad ha sido una etapa importante debido a que es posible obtener representaciones de menor dimensión que sean más robustas e interpretables. En el escenario actual de abundancia de datos, esta fase cobra mayor relevancia porque nos permite enfocar el análisis en aquellos que sean más importantes, optimizando así los recursos disponibles. Un buen ejemplo de esto es el análisis de fuentes textuales, la indexación de semántica latente (en inglés,

ISSN 1135-5948. DOI 10.26342/2018-61-31

LSI) y la modelación probabilística de tópicos.

El modelado probabilístico de tópicos es una técnica que ha demostrado ser útil para la extracción de información semántica de fuentes textuales. Este tipo de modelos se conciben como mecanismos para abstraer del discurso real una representación más compacta que capture el contenido de los documentos analizados.

En el caso particular de esta aplicación consideramos el modelado de tópicos basado en la técnica de **asignación de Dirichlet latente** (en inglés, LDA), que describe una clase de modelos donde las propiedades semánticas de las palabras y los documentos se expresan en términos de tópicos probabilísticos (Blei 2012, Steyvers y Griffiths, 2007).

Existen escasas o nulas aplicaciones con una interfaz gráfica que usuarios interesados en realizar este tipo de análisis puedan usar directamente. Un ejemplo de aplicación del tipo anterior es **Stanford Topic Modeling Toolbox** (TMT)[1]; sin embargo, actualmente ya no es mantenido. Por otro lado, existen paquetes bastante eficientes para el procesamiento estadístico de lenguaje natural que incluyen esta técnica, como es el caso de **Mallet**[2] (McCallum, 2002); sin embargo, la interacción para realizar cualquier análisis es vía comandos, lo que puede resultar sumamente complejo para el usuario común. Finalmente, existen librerías que pueden ser importadas en programas, como es el caso del mismo **Mallet**, **lda**[3], **Gensim**[4], por mencionar algunas de las más conocidas. Desafortunadamente, estas librerías asumen conocimientos de programación avanzados.

Dado el escenario anterior, nuestra propuesta con **QuarryMeaning** es proporcionar una interfaz transparente para el usuario común que desee realizar modelado de tópicos. Para lograr nuestro cometido, aunado a proveer de la funcionalidad mencionada, también incluimos opciones para filtrar palabras no relevantes con la finalidad de mejorar los resultados obtenidos en el proceso iterativo de entrenar un modelo. Dichas palabras pueden ser propuestas por el usuario, o bien obtenidas de forma automática. Por otro lado, consideramos también el filtrado de palabras condicionando por la etiqueta de partes de la oración que le corresponda en el discurso real.

Este tipo de aplicación, finalmente, puede beneficiar a aquellos usuarios que posean conocimientos básicos sobre la técnica de modelación de tópicos y nulos conocimientos de programación porque no requerirán construir ningún tipo de comando y contarán además con mecanismos para reducir *ruido* en los resultados cuando asumen la tarea de realizar un modelado de tópicos a partir de textos.

2 QuarryMeaning: una aplicación stand-alone

QuarryMeaning es una aplicación stand-alone desarrollada en Python, concretamente con el módulo **Tkinter** y se divide en dos fases: entrenamiento y prueba (véase la Figura 1). En la fase de entrenamiento se define el conjunto de documentos que se usará para construir el modelo, así como los parámetros obligatorios y opcionales para llevar a cabo el análisis. Por otro lado, la fase de prueba requiere también de especificar el conjunto de documentos con el que se probará el desempeño del modelo. A continuación, en las siguientes secciones se ofrece más detalle respecto a cada una de las etapas.

[1] https://nlp.stanford.edu/software/tmt/tmt-0.4/
[2] http://mallet.cs.umass.edu/
[3] https://pypi.python.org/pypi/lda
[4] https://radimrehurek.com/gensim/models/ldamodel.html

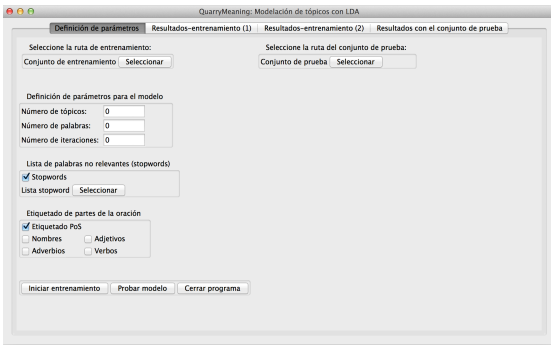

Figura 1. Interfaz del programa

2.1 Entrenamiento

En esta fase, el usuario debe proporcionar la ruta donde se encuentran los documentos que se usarán para la etapa de entrenamiento. Además, deberá especificar los parámetros para realizar el proceso (número de tópicos deseados, número de palabras por tópico y número de iteraciones). Asimismo, si desea filtrar la lista base de palabras no relevantes[5] (en este caso, palabras funcionales) deberá activar la casilla de verificación *stoplist* y, posteriormente, seleccionar el archivo. Del mismo modo, si desea incluir palabras de forma manual en la lista, podrá hacerlo directamente editando el archivo correspondiente. Esta opción es útil cuando una palabra corresponde al dominio analizado, pero aparece en más de un tópico y en todos ellos tiene el mismo significado (palabra no polisémica). Por ejemplo, palabras como *enfermedad, paciente, síndrome* en diferentes subáreas de la medicina tienen el mismo significado, así, si aparecen en varios tópicos correspondientes a sub-áreas como la oftalmología, ginecología, etc., pueden ser filtradas.

Existe también la opción de contrastar el conjunto de entrenamiento con un corpus de referencia y todas aquellas palabras que resulten no relevantes podrán ser agregadas a la lista (Acosta, Aguilar e Infante, 2015). Otra de las opciones consideradas es la inclusión de determinadas categorías de palabras, por ejemplo, nombres y adjetivos. Este tipo de categorías gramaticales son las que más se utilizan para construir términos en un dominio específico por lo que esperamos sean las que tengan el valor semántico más alto. Aunado a estas dos categorías, también se pueden seleccionar verbos y adverbios. Cabe mencionar que cuando se activa la casilla del etiquetado de partes de la oración, éste se realiza con el etiquetador TreeTagger para el español (Schmid, 1995).

Una vez especificados los parámetros y activadas las opciones de análisis deseadas, podemos iniciar el entrenamiento del modelo, para ello se utiliza la librería Python lda[6]. Esto generará un despliegue visual para los primeros 10 tópicos en la forma de nube de palabras[7], lo que facilitará el análisis de los mismos (Figura 2):

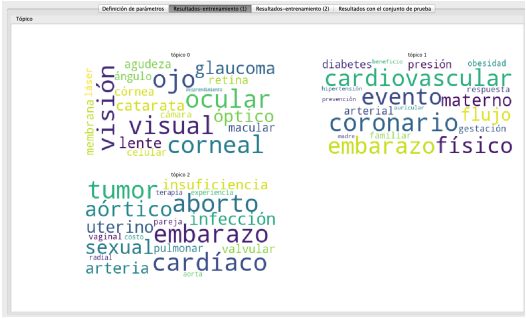

Figura 2. Despliegue visual de tópicos

Si el número de tópicos definido es mayor que 10, tendrá la opción de grabar las imágenes en la carpeta que seleccione para su revisión posterior. Finalmente, podrá ver la distribución de tópicos y el tópico más probable para cada documento dentro de la aplicación, sin embargo, si se han definido muchos tópicos será más apropiado enviar los resultados a un archivo con formato *.csv*, opción disponible también en el programa.

2.2 Prueba

La fase de prueba consiste en procesar los documentos correspondientes al conjunto de prueba con el modelo ajustado y explorar el desempeño observando la categoría predicha

[5] La stoplist base fue seleccionada del módulo NLTK.

[6] https://pypi.python.org/pypi/lda

[7] https://pypi.python.org/pypi/wordcloud

como la más probable (tópico principal). Del mismo modo que la etapa de entrenamiento, es posible grabar como .csv la distribución de tópicos y el tópico principal para cada uno de los documentos de prueba.

3 Evaluación y resultados

Cuando se analiza un método estadístico, es útil probarlo con un caso muy simple donde *a priori* se conozca la respuesta correcta, en este caso, el número de tópicos y así observar si el algoritmo puede distinguir correctamente los tópicos. Lo anterior nos sirve para probar la eficiencia del modelo. En este sentido, probamos con un conjunto de 245 artículos en español de tres sub-áreas médicas: oftalmología, ginecología y cardiología, obtenidas de **Scielo**[8] en español.

De ese conjunto, 221 artículos se utilizaron para entrenar el modelo y el resto para probar su desempeño. Los resultados obtenidos considerando el filtrado de palabras no relevantes obtenidas con el enfoque de comparación de corpus y la lista base, considerando además, sólo nombres y adjetivos, así como el filtrado de palabras no polisémicas pero presentes en más de un tópico. Considerando todo esto, se logra una precisión del 100% en la clasificación de los artículos del conjunto de prueba.

4 Conclusiones y trabajo a futuro

Presentamos **QuarryMeaning**, una aplicación *stand-alone* que facilita el entrenamiento y prueba de un modelo de tópicos a usuarios que requieran realizar este tipo de análisis y no cuenten con el conocimiento ni la habilidad en programación.

Los resultados de la evaluación realizada a la herramienta mostraron que es posible generar un modelo de tópicos con un buen desempeño a partir de seleccionar conjuntos de documentos representativos para entrenarlo, no necesariamente enormes cantidades de ellos. Así, en un proceso de entrenamiento iterativo, las diferentes opciones disponibles permiten la obtención de mejores resultados.

Como trabajo futuro, deseamos adaptar la herramienta para trabajar con otros lenguajes, por ejemplo, el inglés. Además, de incluir un módulo para el análisis del número adecuado de tópicos que sirva de soporte para el usuario al momento de decidir cuántos tópicos considerar.

Bibliografía

Acosta, O., C. Aguilar y T. Infante. 2015. Recognition of Terms in Spanish by Applying a Contrastive Approach. *Linguamatica*, *7*(2): 19-34.

Arun, R., V. Suresh, C. Madhavan y M. Murthy. 2010. On finding the natural number of topics with latent dirichlet allocation: Some observations. En Zaki, M., J. Xu Yu, B. Ravindran y V. Pudi (eds.), *Advances in Knowledge Discovery and Data Mining*. Berlin, Springer: 391-402.

Blei, D. 2012. Probabilistic topic models. *Communications of the ACM*, 55(4): 77-84.

McCallum, A. K. (2002). Mallet: A machine learning for language toolkit. University of Massachusetts, Amherst, Mass., USA: http://mallet.cs.umass.edu/.

Steyvers, M., y T. Griffiths. 2007. Probabilistic topic models. En Landauer, T., D. McNamara, S. Dennis y W. Kintsch (eds.), *Handbook of latent semantic analysis*, Routledge, Oxford, UK: 427-448.

Schmid, H. 1994. Treetagger a language independent part-of-speech tagger. En *Proceedings of the International Conference on New Methods in Language Processing*, Manchester, UK: 44-49.

[8] www.scielo.org